

## User's guide through Pulse Shape Analysis routines

Petar Žugec  
(pzugec@phy.hr)

Department of Physics, Faculty of Science, University of Zagreb

16. December 2014.

- **Universality of routines** (in order to be applicable to the wide range of detectors and signals)
- **Simplicity and directness** (of course, only as much as possible)
- **Minimal number of input parameters** (again, a general guideline more than a rule set in stone)

For example, in order to extract the energy deposited in the detector, between the signal integration and the so called 'trapezoid reconstruction', integration has a clear advantage. Every pulse has an area that can be integrated (and quite simply so), while trapezoid reconstruction explicitly assumes the exponential pulse shape and requires 3 additional parameters (which are not even mutually independent) that need to be manually adjusted. Modular structure of the program does allow to include multiple procedures, but the more general ones will be preferred over more specific ones (which will, in fact, be avoided) as long as the specific one is not shown to be clearly superior at a given task, for a given detector or a signal type.

## How do we handle the new signals, detectors or challenges?

Rather improve, generalize and build upon the existing procedures than immediately introduce the new convoluted ones (of course, only as long as this is possible or efficient). The purpose of the new Pulse Shape Analysis framework is not to be compendium of all signal analysis procedures known to the mankind, but quite the contrary – to be as compact as possible.

**Small minus:** if the changes in the existing procedures require some new UserInput parameters, other users also need to take note of this (though, introducing the brand new procedure could require even more parameters to be added, which would further increase the complexity of the UserInput file, while we are trying to keep it to a minimum). However, if the user was happy with the old version of the procedure, there is always such a value for the newly added parameter which would render the latest updates ineffective. And this value is usually trivial, meaning that no time consuming optimization is needed just to keep the things as they were. For example, the minimal expected width for a  $\gamma$ -flash pulse is used to handle the false  $\gamma$ -flash pulses masquerading as real ones. If a given detector didn't suffer from this problem in a first place, one would just set the value 0 (the 'trivial' case), since all pulses would pass this check.

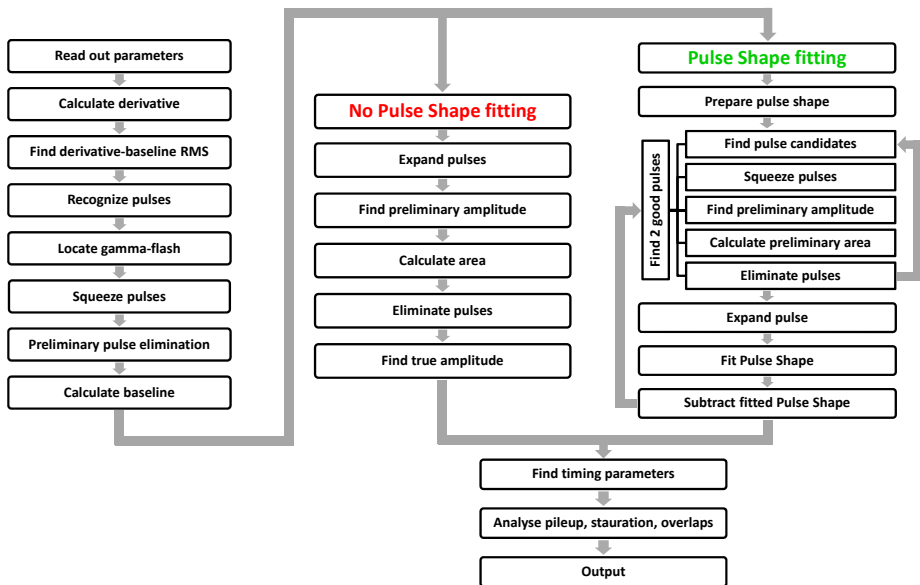
**Big plus:** all detectors can immediately enjoy the benefits of the updated procedure.

# The fall of simplicity

The simplicity of the code (though not of the ideas behind it) was, in fact, brutally sacrificed on several occasions, in order to improve the speed of the program, which has recently become a serious issue. The first, very simple algorithms that were used for the derivative calculation, adaptive baseline calculation and Pulse Shape fitting (all explained later), were all replaced by the less transparent recursive algorithms. It is to be noted that this was done without any loss in precision, since these algorithms are not approximative, but more computationally efficient, yielding exactly the same results as the simpler procedures. Of course, the increased speed, i.e. the reduced number of operations to be performed comes at a price of a more complex mathematical background.

In short, the simplest versions of the listed procedures employ nested for-loops, requiring  $O(N^2)$  operations, where  $N$  is the number of points in the movie. All these were replaced by the algorithms requiring only  $O(N)$  or  $O(N \log N)$  operations, which is an immense improvement, especially if one considers that we are frequently dealing with movies of  $N = 8 \times 10^6$  points!

# Very basic program flowchart



- The analysis procedures are described in the order they are performed, following the flowchart from the last page. This allows to gain an insight into why certain procedures are performed at the certain point, or why they are even necessary.
- Afterwards, the general points are given for the optimization of the UserInput parameters, when starting 'from scratch'.
- At the end, very useful practical advices are listed – gathered from the users' experience – that can help in understanding the effects of UserInput parameters and the results of the analysis procedures, and also in avoiding some common mistakes.

Reading out parameters

Derivative calculation

Derivative-baseline RMS

Pulse recognition

Locating  $\gamma$ -flash

Squeezing the pulses

Preliminary pulse elimination

Baseline calculation

Pulse expansion

Amplitude and area

Pulse elimination

Pulse Shape fitting

Timing properties

Pileup and saturation

Optimization of parameters

Practical advices

# UserInput file

The first part of the UserInput file contains various comments, mostly the short description of parameters. In this section anything can be added or removed. The start of the actual parameter lines is signaled by the first line starting with '~' character. The parameter lines may be commented with '#' character, which enables to have multiple detector settings (for the same detector) present in the file, for easy switching between them (by commenting or uncommenting given lines).

DETECTOR NAME

DETECTOR NUMBER

STEP SIZE

MIXED POLARITY

EXPAND PULSES

TIME LIMIT

G-FLASH OPTION

G-FLASH THRESHOLD

G-FLASH MIN\_WIDTH

G-FLASH WINDOW

BASELINE OPTION

BASELINE FILTER

AMPLITUDE OPTION

AMPLITUDE THRESHOLD

AREA/AMP. LOW THR.

AREA/AMP. HIGH THR.

SIGNAL WIDTH LOW THR.

SIGNAL WIDTH HIGH THR.

NUMBER OF PULSE SHAPES

PULSE SHAPE ADDRESS

# Reading out parameters

The signal to be analyzed and the parameters related to it (length, sampling rate, polarity...) are sent through the header of the central PSAnalysis function. Some of the parameters related to the signal analysis are hard-coded at the beginning of this function – e.g. location of the UserInput file, baseline RMS multiplication factor for pulse recognition, constant fraction for pulse timing and so on.

The type of detector being analyzed is identified by the 4-character **DETECTOR NAME** parameter (SILI, C6D6, MGAS...). If there are multiple detectors of the same type (e.g. more than one C6D6 detector), each detector may be assigned its unique analysis parameters based on the **DETECTOR NUMBER** parameter. The default settings for all detectors of the given type may also be defined under **DETECTOR NUMBER=0**. For example, if there is a line defined for C6D6-1 detector, these specific settings will immediately be used for the detector 1. But if there is no specific line for a given detector number, the default C6D6-0 settings will be used.

# Derivative calculation

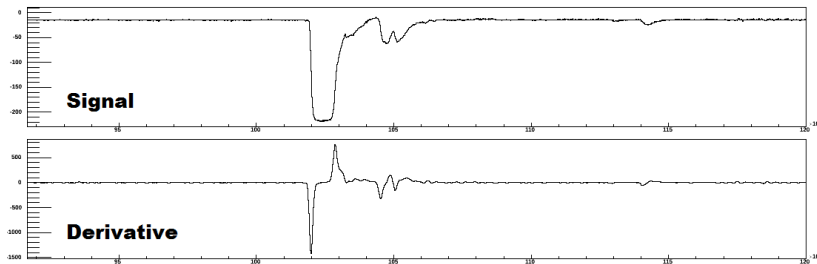
The signal derivative is used for the pulse recognition. However, the derivative needs to be smoothed, since – in general – integrating the signal reduces the noise, while differentiating it further increases the noise. Therefore, for the derivative  $\delta_i$  at  $i$ -th point we do not use the formal definition, but the following one:

$$\delta_i(N) = \sum_{k=1}^N s_{i+k} - \sum_{k=1}^N s_{i-k}$$

which takes advantage of integrating the signal  $s$  at both sides of a selected point, thus reducing the noise and smoothing the final result. The number of points  $N$  to be taken for integration is given by the **STEP SIZE** parameter. The step size is to be set in nanoseconds. The equivalent number of points  $N$  is calculated automatically from the signal sampling rate. The optimal **STEP SIZE** is roughly determined by the pulse risetime and should be manually adjusted until achieving the best signal-to-noise ratio in the calculated derivative.

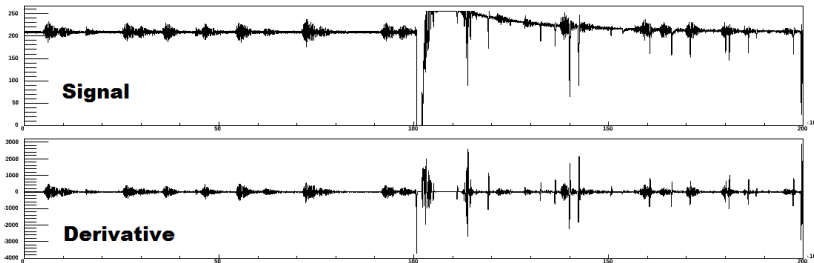
# Derivative-baseline RMS

Pulses are recognized in the signal when their derivative crosses a certain threshold value. This threshold is determined from the RMS (root mean square) of the those portions of derivative that correspond to the signal baseline. The threshold has been selected as  $3.5 \times \text{RMS}$ .



In case the default RMS multiplier of 3.5 is not good enough, the users may select their own value. It is to be set under the **STEP SIZE**, in a way: *step\_size/rms\_multiplier*, e.g. 100/2.5. The numbers are separated by '/' character, without spaces! This value **does not need** to be set.

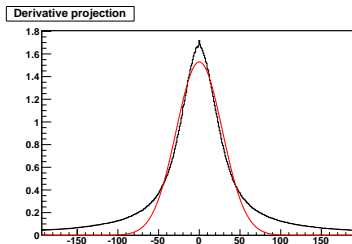
In case of the very clean signals the baseline might be analyzed from a given number of presamples. However, if the presamples-portion of the signal is not as clean, we do not want to absorb the excessive noise or even the early pulses into the baseline RMS, since this would increase the threshold, preventing the recognition of lower pulses. On the other hand, there are cases when the number of presamples is not even well defined, which makes this kind of approach completely inappropriate.



# Derivative-baseline RMS

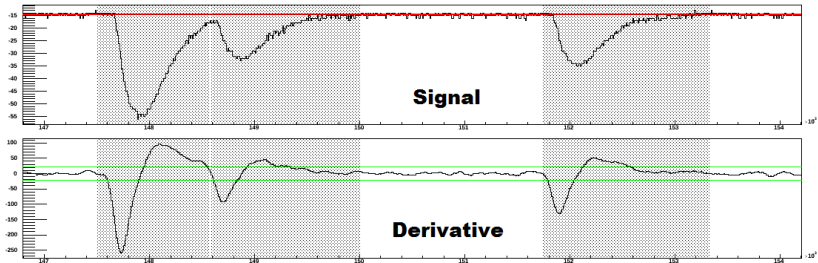
For these reasons we use the special procedure for determining baseline, which completely circumvents the dependence on presamples. First, the whole derivate is projected onto  $y$ -axis, i.e. the histogram is filled with all calculated derivative points. One would expect that the points from the derivative baseline would group around 0, forming the Gaussian with the RMS we are looking for. (Notice that this procedure makes use of all baseline points from a given signal, not only those from the presamples!) However, the points from the actual pulses tend to create the long tails, which we want to exclude from RMS determination.

Therefore, we make few technical tricks to better pronounce the central peak and apply a weighted fitting so as to dominantly absorb the central part of the distribution, corresponding to the actual baseline. From the fitted Gaussian we extract the sought RMS.



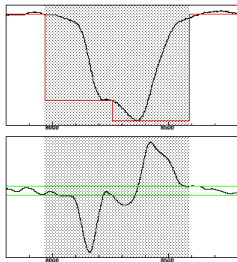
# Pulse recognition

Many procedures depend on the assumption of the pulse polarity so we treat all pulses as negative. This means that if the pulses in the originally delivered signal are positive, we invert the polarity before any further analysis. In order for a negative pulse to be recognized, its derivative – in principle – must make 4 threshold crossings: lower-lower-upper-upper. When the **STEP SIZE** for calculating the derivative is optimized, the recognition of the pileup is performed without a problem.

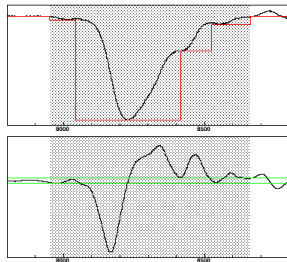


# Pulse recognition

The principle of 4 threshold crossings has been relaxed in order to better recognize the smallest pulses and resolve the pileups that would not be resolved otherwise. Accordingly, it is now sufficient that the derivative triggers either of the two thresholds. However, if both thresholds are crossed in the order lower-upper, this is still recognized as a single pulse, instead of two. These are the threshold crossing possibilities that mark the presence of the pulse: lower-lower (without subsequent upper crossing), upper-upper (without previous lower crossing), lower-lower-upper-upper.



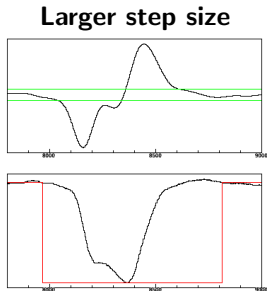
← Signal →



← Derivative →

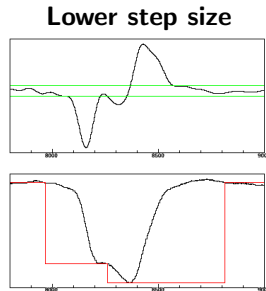
# Pulse recognition

If close pileups have not been resolved at first, the step size for calculating the derivative should be reduced, until the dip between the pulses is able to trigger at least one of the thresholds. Reducing the step size may even be done at a price of reducing the signal-to-noise ratio in the derivative, which is the signature of the derivative becoming more locally sensitive. In fact, reducing the step size may be continued as long as there are no losses in the recognition of 'regular' pulses, but not beyond this point!



← Derivative →

← Signal →

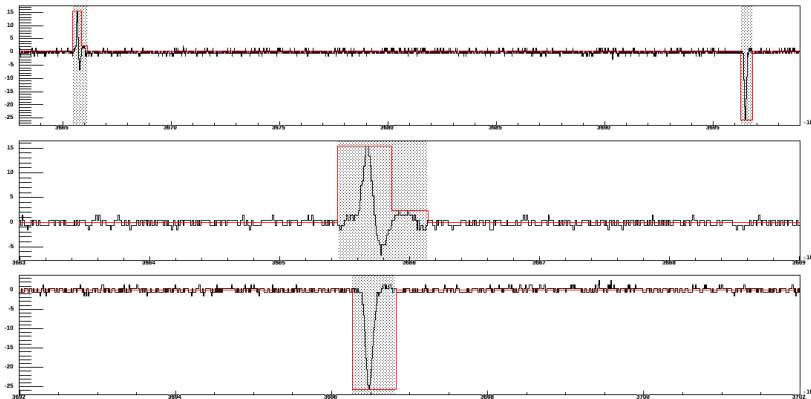


# Pulse recognition (multiple polarities)

Some detectors exhibit pulses of both polarities, i.e. both negative and positive. In this case one may activate the **MIXED POLARITY** option. The program will then search for the negative pulses in the regular derivative and for the positive pulses in the inverted derivative (multiplied by -1). As a result, the reported positions of the positive and negative pulses will most often overlap. This means that alongside every real pulse, one or more false pulses may be returned. The false pulses are then eliminated based on the set of conditions from the UserInput file. Setting **AMPLITUDE THRESHOLD** and **AREA/AMP. LOW THR.** to at least 0 (as opposed to negative values) is very efficient at this elimination of false pulses, since they will often have an amplitude and/or area-amplitude ratio of the 'wrong' sign. It is extremely important to set these elimination conditions appropriately if the **MIXED POLARITY=1** option is used alongside the Pulse Shape fitting (**AMPLITUDE OPTION=2**; explained later), which repeats the search for pulses upon completing each Pulse Shape fit. Since the renewed search starts from the end of a previous pulse, the recognition of following pulse(s) in line may be affected by the acceptance (i.e. the inadequate rejection) of false pulses!

# Pulse recognition (multiple polarities)

Ideally, the multiple-polarities procedure would be expected to produce the same result as the single-polarity procedure when applied to the single-polarity pulses. However, since it may affect the search for regular pulses and/or admit the false pulses, the **MIXED POLARITY** option should be avoided if the detector does not exhibit multiple polarities!



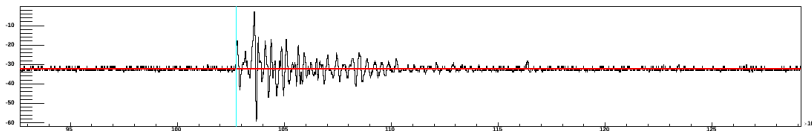
# Locating $\gamma$ -flash

$\gamma$ -flash is recognized as the first pulse satisfying a certain set of conditions. It may be recognized as the first pulse crossing a given threshold (**G-FLASH OPTION=0**) or the first pulse going into saturation (**G-FLASH OPTION=1**). There is a special treatment for oscillatory  $\gamma$ -flash (**G-FLASH OPTION=2**), explained later. In case of the threshold crossing option, **G-FLASH THRESHOLD** must be selected (in channels). If a given detectors suffers from the occurrence of false  $\gamma$ -flashes – i.e. the pulses preceding the actual  $\gamma$ -flash, which are able to cross the selected threshold – one may additionally set the **G-FLASH MIN\_WIDTH**, which is the minimal expected width for the real  $\gamma$ -flash. In most cases, this should be enough to discriminate the real  $\gamma$ -flash from the false ones.

$\gamma$ -flash is located before calculating the overall signal baseline, but the procedure has its own independent baseline search! This means that the **G-FLASH THRESHOLD** is indeed to be given in the number of channels relative to the baseline.

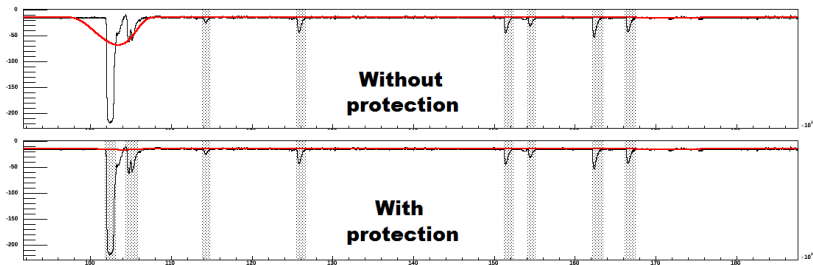
# Locating $\gamma$ -flash

In case of the wildly oscillatory  $\gamma$ -flash (such that oscillations cross the baseline), a special procedure is used (**G-FLASH OPTION=2**). After the first threshold crossing, the maximum of the signal is sought inside the next half (!) of the **G-FLASH MIN\_WIDTH** window. This maximum is considered the top of the  $\gamma$ -flash, envelope of the oscillations is determined inside the half (!) of **G-FLASH MIN\_WIDTH** starting from its top and the constant fraction timing is performed to find the starting time. Note that this time the minimal width is not checked, but used for manipulation!



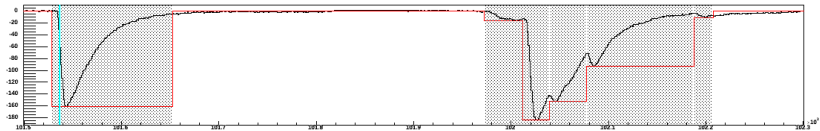
The  $\gamma$ -flash arrival time is determined by the constant fraction triggering, using the default value of 0.3 (that is, 30% of the amplitude). In case a different value is required, it may be set under the **G-FLASH OPTION** as: *flash\_option/constant\_fraction*, e.g. 0/0.1. The numbers are separated by '/' character, without spaces! This value **does not need** to be set.

Very important later procedure is the elimination of false pulses. However, the conditions that would keep only regular pulses would most often eliminate the  $\gamma$ -flash pulse, which must not happen because the  $\gamma$ -flash surroundings are very important for the later baseline calculation! For this reason there is a **G-FLASH WINDOW** parameter, which is the length of time after the start of  $\gamma$ -flash, where all the pulses are protected against elimination (if at least a part of the pulse is within this window). Still,  $\gamma$ -flash **may** (and often **should**) be eliminated during the Pulse Shape fitting (explained later), after the baseline has been calculated.

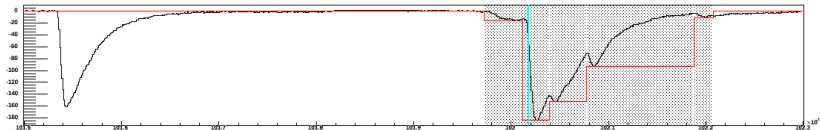


# Locating $\gamma$ -flash (example)

At the first go, 6 separate pulses are recognized in this movie, and based only on the threshold crossing the first one is mistaken for the  $\gamma$ -flash:



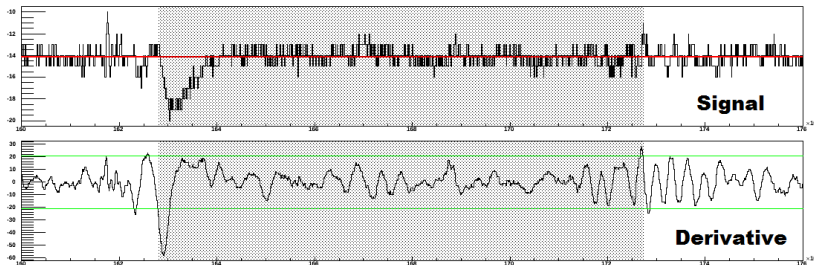
After including the minimal expected width (which is applied to the whole chunk of signal above the baseline, even though the parts of it may have been recognized as separate pulses!) and setting the severe elimination conditions (just for illustration), the second pulse is recognized as a real  $\gamma$ -flash and is protected against elimination (while the first one is indeed eliminated, not actually being the  $\gamma$ -flash):



In case the waveform (corresponding to a single neutron bunch) is separated into multiple movies by the zero-suppression, the movies are delivered to the analysis procedure in chronological order, one by one. Thus, if  $\gamma$ -flash is located in some of the later movies, the earlier ones will not have the information about the  $\gamma$ -flash arrival! For these movies the  $\gamma$ -flash is assigned a default value: **0**! (All movies following the  $\gamma$ -flash will share the real  $\gamma$ -flash value.) **Therefore, care must be taken during the data analysis not to calculate the time of flight without checking the  $\gamma$ -flash value!** If the counts preceding the  $\gamma$ -flash are of no interest, they should simply be rejected from analysis. Otherwise, the real value of the  $\gamma$ -flash (for those counts that have the reported value 0) may be found by searching the root-files for the first count with the same bunch number (i.e. belonging to the same neutron bunch) and the non-zero  $\gamma$ -flash value.

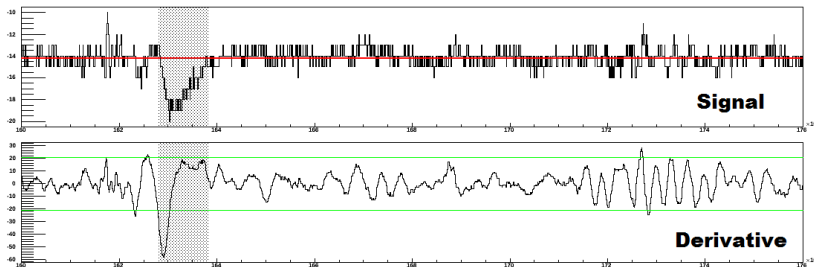
Sometimes, during the recognition of lowest pulses, the pulse will not be 'closed' by itself, but by the noise accidentally triggering the derivative threshold. Thus, the length of the pulse will be much too wide and the pulse might be eliminated by the subsequent elimination procedure (explained later). In order to rectify this, the pulse length is renormalized by 'squeezing' it.

## Before squeezing



If the pulse was recognized by the lower-lower-upper-upper crossing, the squeezing starts by finding the position of the minimal point in the signal (which, before the baseline correction, corresponds at least approximately to the amplitude position). In case of lower-lower or upper-upper crossing, the appropriate extremum is found in the derivative. Then the zero-crossing in the derivative is sought again at both sides of this point, thus redefining the pulse boundaries. This procedure is completely autonomous and does not require any input from users.

## After squeezing



# Preliminary pulse elimination

In searching for pulses in the derivative, the lower derivative-threshold is preferred over the higher one, in order not to miss the lower pulses. This hold only as long as 'lower' and 'higher' alternatives are both in the vicinity of the optimal threshold. In this, our paradigm is: **first recognize everything (and then some), and then eliminate the false pulses.**

Preliminary pulse elimination is performed only based on the pulses' widths (**SIGNAL WIDTH LOW THR.** and **SIGNAL WIDTH HIGH THR.**), since their amplitudes and areas have not yet been determined. This procedure is extremely important since the threshold low enough to be triggered by the lowest pulses will from time to time be accidentally triggered by the noise. Thus the large chunks of the baseline may at first be recognized as potential pulses. These false pulses must be eliminated as well as possible, as soon as possible, since the later baseline calculation and some other procedures depend on the reported pulses. In addition, all pulse candidates will later be subjected to the pulse shape analysis procedures, which implies increased time consumption if the false pulses must be analyzed alongside the real ones.

# Baseline calculation

Baseline is treated differently below and above the **TIME LIMIT**, which is set by the user. **TIME LIMIT** defines the range where the effect of the  $\gamma$ -flash (such as baseline oscillation or undershot) is still present and the adaptive baseline calculation is required. Above **TIME LIMIT** constant baseline is calculated, even if the **TIME LIMIT** is in the middle of a movie. The method for calculating the baseline below **TIME LIMIT** is selected via **BASELINE OPTION** parameter. **BASELINE OPTION**=0 means constant baseline, which gives the same result as setting the **TIME LIMIT** below the start of the movie. **BASELINE OPTION**=1 is presently the best available adaptive procedure, based on weighted signal averaging.

**BASELINE OPTION**=2 is the procedure from M. Morháč, NIM A 600, 478 (2009), using the decreasing clipping window, smoothing algorithm and adaptive peak regions (see the reference for explanations). It was considered the best option before the certain improvements were introduced to the **BASELINE OPTION**=1 procedure. However, **BASELINE OPTION**=2 is very time consuming, so it is not the recommended option anymore.

Special **BASELINE OPTIONS** 3–6 are also available (explained later).

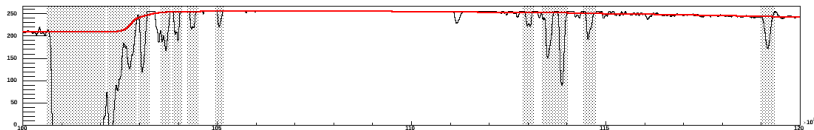
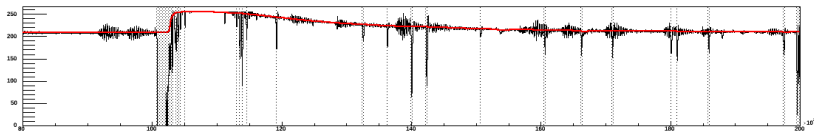
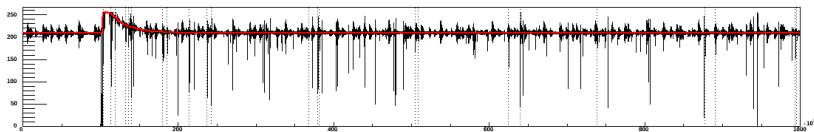
# Baseline calculation

Constant baseline is calculated from the points between the reported pulses, therefore not relying on the (existence of) signal presamples.

**BASELINE OPTION**=1 calculates the adaptive baseline  $B$  as the weighted moving average of the signal  $s$ :

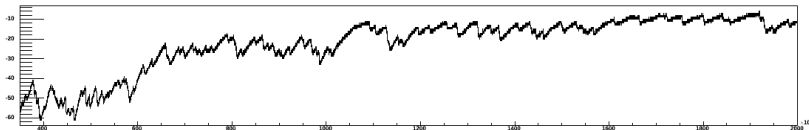
$$B_i = \frac{\sum_{j=i-N/2}^{i+N/2} s_j w_j \{1 + \cos[(j-i) \frac{2\pi}{N}]\}}{\sum_{j=i-N/2}^{i+N/2} w_j \{1 + \cos[(j-i) \frac{2\pi}{N}]\}}$$

The weighting kernel is given by  $(1 + \cos)$ -filter, with additional weighting factors  $w$  which are equal to the number of points in the uninterrupted portions of the baseline, between the reported pulses. Total number of points  $N$  taken for averaging around  $i$ -th point is determined by the **BASELINE FILTER** parameter. **BASELINE FILTER** is to be set in nanoseconds, while  $N$  will be automatically calculated from the signal sampling rate. **BASELINE FILTER** should be wide enough to connect the baseline at both sides of a given pulse, which is often (not always!) best adjusted looking at the  $\gamma$ -flash pulse, since it is usually the widest one.



# Baseline calculation (special case)

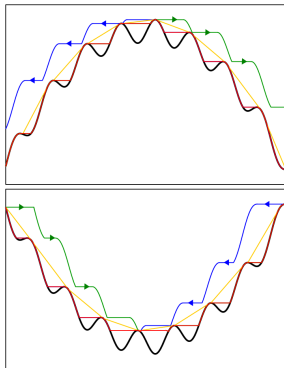
Normally, the calculated baseline goes through the 'bottom' of the signal, cutting through the fluctuations. However, in case of sequential and persistent pileups, the information about the baseline may be almost completely lost, since the 'bottom' of the signal is not restored for a long time. In this case the best, if not the only assumption to make is that the baseline follows – with a certain degree of sensitivity – the dips between the pulses, particularly those that manage to reach most deeply toward the baseline. Therefore, the 'upper envelope' of the signal needs to be calculated (since the pulses are always treated as negative).



# Baseline calculation (special case)

In order to find the signal envelope, the 'moving maximum' is calculated, analogous to the moving average. First, calculation is performed forwards, meaning that for the value at a given point a signal maximum is taken from a certain window **before** this point. Then the maximums are calculated backwards, taking the signal maximum from a window of the same length **after** a given point. Out of two maximums (forward and backward one), a minimal one is taken for a final result. In addition, this envelope that 'slithers' along the signal may be 'tightened' (like pulling on an anchored rope).

## Proof of concept



- Artificial signal
- Forward maximum
- Backward maximum
- Combined maximum
- Tightened maximum

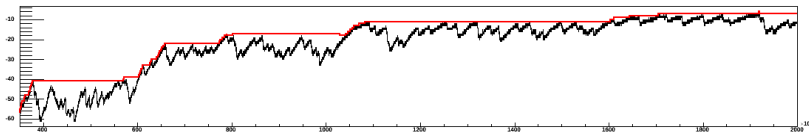
# Baseline calculation (special case)

This procedure is incorporated as a **BASELINE OPTION=3** and **BASELINE OPTION=4**. Option 3 calculates the baseline without tightening, while option 4 does apply tightening. At the end, both results can be smoothed by a  $(1+\cos)$ -filter from **BASELINE OPTION=1** (with all the extra weights taken as  $w_i = 1$ ). Contrary to options 0–2, options 3 and 4 do not depend on the reported pulse positions!

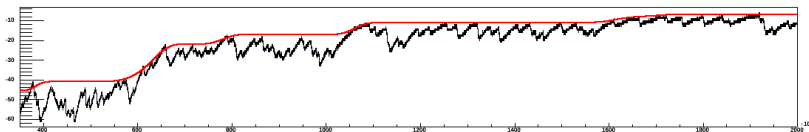
Evidently, alongside the **BASELINE FILTER**, the width of a sliding window for calculating the local maximums must also be set. This is done within the **BASELINE OPTION**, in a way: *baseline\_option/sliding\_window*, where the numbers are separated by '/' character, without spaces! For example: 3/5e5.

By adjusting the sliding window, a greater or weaker local sensitivity to the dips between the pulses may be achieved. The envelope made of local maximums may be exposed by setting the smoothing window to 0. After the envelope has been properly adjusted, only then should the smoothing window be selected. This procedure may be expected not to properly reconstruct the baseline at the start of the  $\gamma$ -flash.

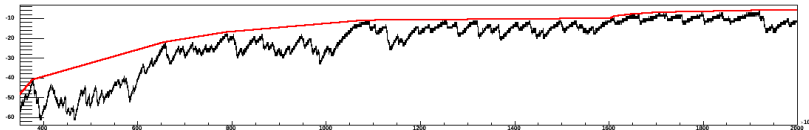
## Option 3 – before smoothing



## Option 3 – after smoothing



## Option 4 – before smoothing



# Baseline calculation (most general case)

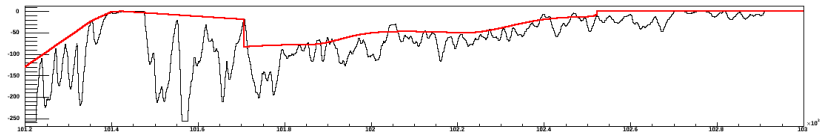
**BASELINE OPTIONS** 5 and 6 allow to use 3 different baseline modes, in this exact order: baseline from maximums (equivalent to options 3 or 4), filtered baseline (equivalent to option 1) and the constant baseline (equivalent to option 0). The difference between options 5 and 6 is the same as between options 3 and 4: the tightening is or is not applied. Option 5 is composed of modes 3-1-0, while option 6 of modes 4-1-0.

Two time limits must be set here. The format for the **TIME LIMIT** is the following: *time\_limit\_1 / time\_limit\_2*, where the numbers are separated by '/' character, without spaces! For example: 2e5/1e6. The sliding window for option 3/4 is set in the usual manner, as a second argument under the **BASELINE OPTION**. The smoothing window for option 3/4 and the baseline filter for option 1 may also be set independently, under the **BASELINE FILTER**. The format is: *baseline\_filter / smoothing\_window*, where the numbers are separated by '/' character, without spaces! For example: 5e4/1e5.

# Baseline calculation (most general case)

At any of the time limits – where baseline modes interchange – the discontinuity in baseline may be expected. However, if the time limit resides within the recognized pulse, this discontinuity would ruin the pulse reconstruction (its shape and area). In this case the time limit is automatically pushed outside the pulse, in order not to destroy the pulse.

In this picture the parameters were intentionally unoptimized in order to illustrate the coexistence of 3 baseline modes. When the baseline parameters are optimized, the discontinuities should be barely visible.



Notice that there is a slight (but not complete!) redundancy among the **BASELINE OPTIONS**. By properly selecting two time limits, options 5/6 may completely revert to either option 3/4 or option 1 or option 0. In other words, options 3/4, 1 and 0 are special cases of options 5/6.

# Pulse expansion

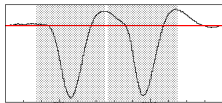
At first, pulses are recognized between the points of first and last (fourth) derivative-threshold crossing. Then the ending edge is further pushed until the derivative reaches 0. Sometimes – especially in case of bipolar pulses, or pulses with the very slow tails – this procedure does not 'close' the pulse properly. In order to deal with this, one may choose to activate the **EXPAND PULSES** option, which shifts the boundaries of the pulse based on what happens in the actual signal, instead of the derivative. As soon as the **EXPAND PULSES**  $\neq 0$  is activated, the starting edge of the signal is manipulated independently of the actual option selected. Therefore, the particular options regard only the ending edge of the pulse.

If the starting point of the pulse is outside the 'main body' of the pulse (where 'main body' is the central part of the pulse, which is entirely above the baseline), then this point is pushed forwards, until it reaches the baseline. Therefore, if the point was within the baseline fluctuations, most of these fluctuations will be kept. If the point was within the earlier undershot (on top of which the pulse started), the point will be pushed forwards to the start of the 'main body'. If the starting point was inside the 'main body', it will be pulled back, to the start of it.

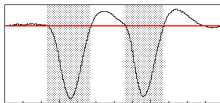
# Pulse expansion

The ending of the pulse may be pulled backwards (option -1) or pushed forwards (option 1) until the signal reaches the baseline. This is crucial for the bipolar pulses, which exhibit a pathology, since the originally found pulses stop at the extremum of the second pole (where derivative is 0). However, this expansion attempt will be stopped in its tracks if there is a next pulse in line blocking it. Since their front edges are adjusted first, the next (real) pulse should not interfere with this (it has already been pushed towards the baseline). But there might be leftover chunks of noise in the undershoot that were not properly eliminated, standing in a way of a further expansion. To overcome this, there is a more insistent option 2 that makes an additional forceful push, trampling over these bothering pulses. This push will be made only if the initial ending point was below the baseline, so that regular pileups will not be trampled.

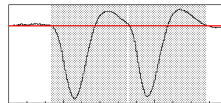
**Option 0**



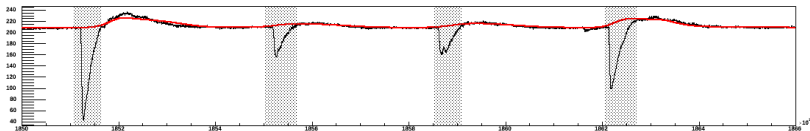
**Option -1**



**Option 1 / 2**

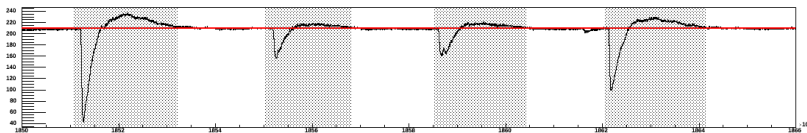


There is also the special **EXPAND PULSES** option -2, designed for the bipolar pulses with the long undershot. In this case one may want to pull back the end of the pulse (option -1) in order for a large portion of negative area from undershot not to interfere with the positive area from under the main part of the pulse. However, if somewhat narrow **BASELINE FILTER** is used – so that baseline could be sensitive to the immediate pulse surroundings – then the baseline will pick up the undershot, since it is no longer part of the pulse. As a consequence, the baseline under the main part of the pulse will bend, 'burrowing' towards the undershot:



# Pulse expansion (special case)

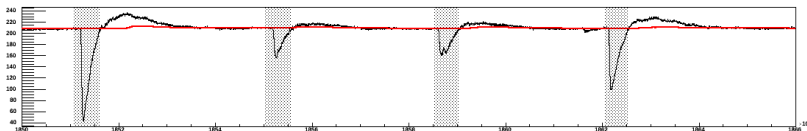
One way to avoid this is to expand the pulses with option 2 (pushing their edges forwards), calculate the baseline (which is now insensitive to the undershoot, since it has become a part of the pulse) and then pull back the edge of the pulse. However, the baseline must be calculated before the first push! So the only way to obtain a flat, locally less sensitive baseline is to use the very wide **BASELINE FILTER**. Afterwards, either the push or pull can be made:



However, such a wide **BASELINE FILTER** may be incompatible with the  $\gamma$ -flash region, which may exhibit a much faster baseline restoration.

# Pulse expansion (special case)

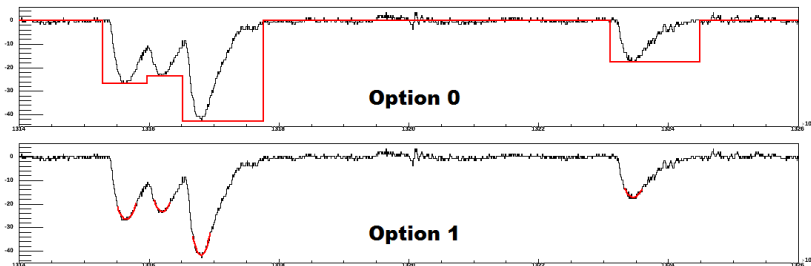
Therefore, after the first baseline calculation (using the wide **BASELINE FILTER**) the pulses' boundaries are pushed forwards and then the baseline is recalculated using the shorter baseline filter. In this way the baseline is made more locally sensitive, but simultaneously blind to the undershoot, which is now inside the pulse boundaries. Once this baseline is found, the pulses' edges may finally be pulled back, with the flat baseline remaining under the their main parts:



In this case two baseline filters must be set: one wider and one shorter. The shorter (i.e. the final) filter is set normally as a **BASELINE FILTER**, while the wider one is set within the **EXPAND PULSES** option. The format is the following: *expand\_option/wide\_filter*, where the numbers are separated by '/' character, without spaces! For example: -2/5e4.

# Finding the amplitude and area

For finding the amplitude of the pulse, several options are available. **AMPLITUDE OPTION=0** implies the search for the highest point of the pulse. **AMPLITUDE OPTION=1** activates the parabolic fitting to the top of the pulse, while **AMPLITUDE OPTION=2** activates the predefined Pulse Shape fitting. In case of the last option, Pulse Shapes need to be provided and both the final amplitude and area will be determined from the fitted pulse. Otherwise, the area under the pulse is calculated by the simple pulse integration.



# Pulse elimination

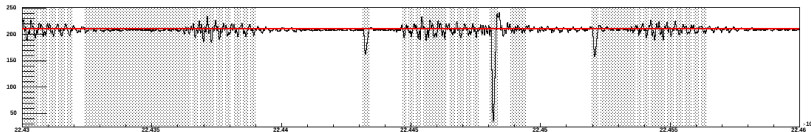
Even if **AMPLITUDE OPTION**  $\neq 0$  is selected, amplitudes and areas are first determined by the simplest procedures – search for the maximum and integration. Based on this results, the pulses are subjected to the elimination algorithm, using the **AMPLITUDE THRESHOLD**, **AREA/AMP. LOW THR.** and **AREA/AMP. HIGH THR.** in order to cut away false or unwanted pulses. The elimination is performed before applying more complex procedures (parabolic or Pulse Shape fitting) in order for these procedures not to spend time on analyzing the pulses that will be rejected anyway.

**SIGNAL WIDTH LOW THR.** and **SIGNAL WIDTH HIGH THR.** are **not** used for elimination at this point! They were 'spent' during the preliminary elimination, which had to be performed before the baseline calculation, which, in turn, had to be performed before the pulse expansion. If these widths were used now, they would have to accommodate both expanded and unexpanded pulses, thus reducing their effectiveness. Therefore, the signal widths should be adjusted before activating **EXPAND PULSES**  $\neq 0$  option (i.e. option 0 should be used).

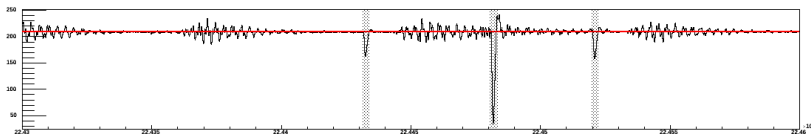
# Pulse elimination

Contrary to the preliminary elimination from few steps back (which may heavily affect the whole subsequent analysis), this one serves to possibly save some processing time and – most importantly – to manage the size of the final root-files by not filling them excessively with the useless data. The final elimination of the useless data **should** (!) be done afterwards, during the later data analysis! For this the additional data from the root-files (e.g. saturation and pileup flag, risetime etc.) may also be used.

**Before elimination:**



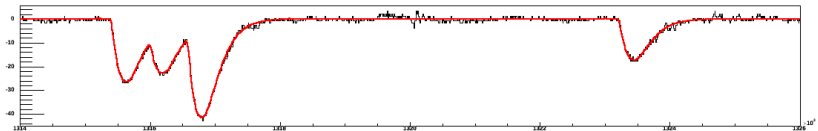
**After elimination:**



Pulse Shape fitting is activated by setting **AMPLITUDE OPTION=2**. The number of different available Pulse Shapes (for different pulse types) is set through the **NUMBER OF PULSE SHAPES** parameter. Under the **PULSE SHAPES ADDRESS** this exact number of addresses for the Pulse Shape files must be given. The files need to contain two columns – the first one determines the timing ( $x$ -coordinate) of points in nanoseconds, the second one gives their heights ( $y$ -coordinate) in arbitrary units. The submitted pulse polarity should correspond to the polarity of the actual pulse. The program will automatically take care of inverting the polarity if necessary. The fitting will proceed fine even if the polarity of the Pulse Shape is wrong but the final amplitudes will then have a wrong sign! The first column defines the Pulse Shape sampling rate. It does not need to be the same as the sampling rate of the signal being analyzed. Program will take care of the synchronization between the two. In addition, for every Pulse Shape the program will produce several refinements (currently 4), by interpolating between the original points.

# Pulse Shape fitting

Fitting proceeds by shifting the Pulse Shape around the (approximate) amplitude position of the analyzed pulse. At each step the the Pulse Shape is adjusted to the analyzed pulse using the least squares procedure. The best position is determined from the minimal reduced  $\chi^2$  of the fits at different steps. At the best position for the original Pulse Shape the interpolated Pulse Shapes are also fitted and the best refinement is again determined from the minimal reduced  $\chi^2$ . The procedure is repeated for every single Pulse Shape submitted. The best fitted result is selected based on the minimum of minimal reduced  $\chi^2$ -s for different Pulse Shapes (i.e. the absolute minimum of all fits performed). In this case both the amplitude and the area under the pulse are determined from the best fitted result. The optimal fitted Pulse Shape is subtracted from the signal before proceeding to the next pulse in line, thus correcting for the pileup!



# Pulse Shape fitting

If the pulse shapes seem to vary with amplitude – so that a single average Pulse Shape is not optimal for fitting all of them – then one may take advantage of possibility to have multiple Pulse Shapes. For example, the separate Pulse Shapes may be provided for low, medium and high pulses.

There is a special option for eliminating the inappropriate fits, since their subtraction could heavily affect the reconstruction of the later pulses. It is performed based on the value similar to  $\chi^2$ , that we call the *discrepancy*:  $\sum_{i=1}^N (s_i - f_i)^2 / (N \times A^2)$ , where  $s$  is the actual (baseline corrected) signal,  $f$  is the fitted Pulse Shape and  $A$  is the amplitude found from the highest point of a signal. The discrepancy is calculated taking into account not only the points from the fit range, but also those extending to the end of the Pulse Shape or to the start of the next pulse. If the discrepancy is larger than a certain goal, the fit is discarded. The goal may be set under the **AMPLITUDE OPTION**, in a way: *amplitude\_option/goal*, where the numbers are separated by '/' character, without spaces! For example: 2/0.5. However, under the normal circumstances this option should **not** be used and the goal should not be set at all! Instead, an attempt at elimination by the regular set of elimination conditions should be made first.

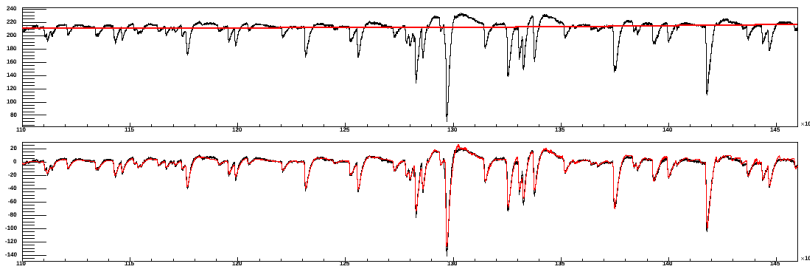
# Pulse Shape fitting

If the Pulse Shape fitting is used alongside the **MIXED POLARITY=1** option, it is extremely important to set the appropriate conditions for the elimination of false pulses. The reason is that after each completed fit the procedure repeats the search for subsequent pulses, starting from the end of the previous one. Therefore, if the false pulse was accepted in place of true one (i.e. it was not properly rejected), its positioning may affect the recognition of the following pulse(s) in line (since the regular and inverted pulse candidates are usually reported as overlapped).

As illustrated in the program flowchart, 2 pulses must be found in each repeated search, due to the possible activation of **EXPAND PULSES=1** option. In this case the second pulse serves to stop the expansion of the first one, thus preventing the second pulse from being trampled and missed in the next search. However, only the first pulse is – at the current point – subjected to the Pulse Shape fitting, while the second one will be considered for fitting during the next search. In case of the Pulse Shape fitting, **EXPAND PULSES=2** option will be automatically degraded to **EXPAND PULSES=1** option, for the very reason of not stepping over the not-yet-recognized pulses.

# Pulse Shape fitting

In case of persistent pileup of pulses with long tails or strong undershoots, the baseline calculated by manipulating the available baseline parameters serves only as the 'primary' baseline. For piled-up pulses the tails of earlier ones serve as the 'secondary' baseline, which may be taken care of by the Pulse Shape fitting. However, the Pulse Shape reconstruction will not be satisfactory if the 'primary' baseline has not been properly determined. Therefore, one may use the (dis)agreement between the actual and reconstructed pulses in order to find the most appropriate 'primary' baseline. When this is done, all fits should 'fall into place'.



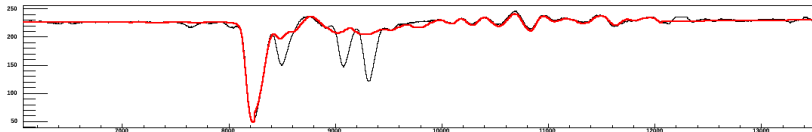
# Pulse Shape fitting ( $\gamma$ -flash)

There is a special option for subtracting the part of the signal immediately after the  $\gamma$ -flash, in order to restore the low time of flight (i.e. the high neutron energy) pulses. Evidently, it is meaningful only if the  $\gamma$ -flash rebound is very consistent between separate neutron bunches. As opposed to the regular Pulse Shape fitting, only one  $\gamma$ -flash Pulse Shape is supported. Its selection is signaled via **NUMBER OF PULSE SHAPES** parameter, in a manner: *regular\_shapes/flash\_shapes*, where the numbers are separated by '/' character, without spaces! Since only one Pulse Shape will be used, the number of  $\gamma$ -flash Pulse Shapes should be 1, for example: 0/1. The address of the file holding the  $\gamma$ -flash Pulse Shape is to be reported as a regular Pulse Shape file – under the **PULSE SHAPES ADDRESS** – but at the end (as the last Pulse Shape address).

At the moment, the  $\gamma$ -flash subtraction procedure is quite particular – adapted to the specific requirements of FIMG detectors. For example, it is assumed that the  $\gamma$ -flash rebound **does not** scale with the height of the initial  $\gamma$ -flash pulse. Therefore, its application possibilities may be somewhat limited, concerning the most general  $\gamma$ -flash behavior.

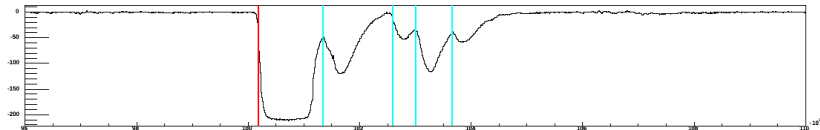
The procedure aligns the  $\gamma$ -flash Pulse Shape to the actual signal by fitting only a portion of a reported Pulse Shape (this portion is set by the user). It has been observed that it is best only to fit the leading edge of the initial pulse. Then the Pulse Shape is directly subtracted from the signal, without scaling it for the multiplicative factor obtained by fitting. However, the fitted part of the Pulse Shape will, in fact, be scaled in the graphical user interface, in order to facilitate the visual inspection of the quality of the fit.

The format of the Pulse Shape file is basically the same as for the regular Pulse Shapes (two columns: one is real time, the other is signal height, but this time in absolute channel units!), with one exception. In the **first line** two numbers must be given (in time units), which define the lower and upper bound for the fitted portion of the Pulse Shape. These numbers are the absolute time-coordinates from the reported Pulse Shape.



# Timing properties

The principle of finding the pulses' timing properties is quite simple and is equivalent to the constant fraction discrimination, i.e. the timing is determined from signal crossing the fixed fraction of pulse's amplitude. For finding the **arrival time** the fraction of 30% is used. The **FWHM** (full width at half maximum) and **FWTM** (full width at tenth maximum) are also calculated in the same manner, by finding the moments of signal crossing 50% or 10% of the amplitude, respectively, at both the leading and falling edge of the pulse. In addition, the **risetime** is calculated as a time needed for a leading edge to rise from 10% to 90% of the amplitude. These numbers may serve as excellent restrictions for the data reduction during the later data analysis. The **peak moment** is also reported. It is found either as the moment of the first highest point, the vertex position of a fitted parabola or the peak position of the fitted Pulse Shape (depending on the selected **AMPLITUDE OPTION**).



Intense pileup may not only affect the amplitude and area determination if the Pulse Shape fitting is not used, but also – to a certain degree – the timing of pulses. Since the arrival moment is determined by signal crossing the constant fraction of pulse's amplitude, if the pulse does not start from a 'true baseline' (where 'true baseline' includes the tails of previous pulses) then the error in finding the 'true' amplitude will also translate into the erroneous timing, affecting the timing resolution. In addition, if the pulse is 'stuck between' the two neighboring pulses – and just barely resolved from them – it will be assigned a very narrow analysis-range width, which calculated risetime and FWHM/FWTM will never exceed, thus becoming unreliable! Without giving any additional information about the expected pulses, no procedure can reconstruct the 'true' parameters, once the pulse has been 'drowned' by and 'dissolved' into its environment. Fortunately, there **is** a way of providing this information, in a form of a Pulse Shape!

# Timing properties (the pileup effect)

Even if the Pulse Shape fitting is used, the risetime and FWHM/FWTM reported in the root-files will be obtained from the analysis of a true pulse, **not** from the Pulse Shape! The reason is evident: these parameters are always the same for a given Pulse Shape, and the users providing the Pulse Shape may always determine them by themselves. Therefore, reporting the same numbers for all pulses would be completely redundant!

Regarding the most important timing parameter – the 'true' arrival time – it may always be easily reconstructed from a reported peak moment! Since (in case of the Pulse Shape fitting) the peak moment is determined from a peak position of a fitted Pulse Shape, the 'Pulse Shape arrival time' may be calculated as:

$$\text{arrival time} = \text{peak moment} - \Delta t$$

where  $\Delta t$  is the time difference between the peak position of a given Pulse Shape and the moment of a Pulse Shape crossing the selected fraction of its amplitude. This value is always fixed (for a given Pulse Shape) and users may find it by the most simplistic analysis of their own Pulse Shape(s).

# Pileup and saturation analysis

Whether the pulse is affected by the pileup is determined from its first and last point. If these points are within certain boundaries (which are found from RMS of the noise in the baseline and are set to  $\pm 5 \times \text{RMS}$ ), then no pileup is considered to have occurred. This is checked separately at the start and the end of the pulse, and is reported through the *pileup\_front* and *pileup\_rear* parameters in the final root-files. Both of these may take on one of three values: -1, 0, 1. Value 0 means no pileup at a given end of the pulse. Value 1 implies that the pulse starts/ends above the threshold ('normal' pileup, meaning the excessively negative starting/ending point for negative pulses), while value -1 implies the start/end below the threshold (for example, from the undershoot of the previous pulse). In case of Pulse Shape fitting, the pileup status is reported for the original pulses, not the ones that may have been corrected for the pileup by the subtraction of the fitted Pulse Shape!

Finally, the saturation is determined simply by checking if at any point the pulse reaches the minimal or maximal channel supported by digitizers (e.g. 0 or 255).



# Optimizing the UserInput parameters



When starting the parameters optimization 'from scratch', this is how the initial settings should look like:

DETECTOR NAME		DETECTOR NUMBER	STEP SIZE	MIXED POLARITY	EXPAND PULSES
(appropriate)		(appropriate)	(anything)	0	0
TIME LIMIT	G-FLASH OPTION	G-FLASH THRESHOLD	G-FLASH MIN_WIDTH	G-FLASH WINDOW	
0	0	1e10	1e10	0	
BASELINE OPTION	BASELINE FILTER	AMPLITUDE OPTION	AMPLITUDE THRESHOLD	AREA/AMP. LOW THR.	AREA/AMP. HIGH THR.
0	(anything)	0	-1e10	-1e10	1e10
SIGNAL WIDTH LOW THR.		SIGNAL WIDTH HIGH THR.	NUMBER OF PULSE SHAPES	PULSE SHAPE ADDRESS	
0		1e10	1	(dummy line)	

- Optimization starts with all elimination conditions completely relaxed and most of the 'extra' options rendered inactive or ineffective.
- First, the **STEP SIZE** needs to be optimized so that the best pulse recognition is achieved. This means that all clear pulses and as much of the pileup should be resolved. At this point, recognizing the chunks of noise as pulse candidates is perfectly fine.
- When this is achieved for regular pulses, the **MIXED POLARITY** option may be activated, if necessary. The inverted pulses will be recognized in parallel with regular ones and will often be overlapped (this overlaps should later be removed by the elimination conditions). If necessary, **STEP SIZE** should be readjusted so that the best recognition of both polarities is achieved.
- Now the **SIGNAL WIDTH LOW THR.** and **SIGNAL WIDTH HIGH THR.** should be adjusted so as to reject most of the false pulses (e.g. chunks of noise). However, the conditions must not be so severe that any of the the real pules are discarded.

- The optimization of signal (i.e. pulse) width limits must be done before activating the **EXPAND PULSES** option, because these limits are used for the elimination only once, and this is before expanding the pulses! At this point the portions of  $\gamma$ -flash that are very different in width than usual pulses may also be eliminated, since later they will be explicitly protected against elimination. In other words, the width limits (and other elimination conditions) are not restricted by the 'quirks' of the  $\gamma$ -flash.
- Now the  $\gamma$ -flash recognition options should be adjusted, including the **G-FLASH OPTION, G-FLASH THRESHOLD, G-FLASH MIN\_WIDTH**. This should be pretty straightforward.
- The baseline parameters should be adjusted next. These include the **BASELINE OPTION, BASELINE FILTER, TIME LIMIT** and **G-FLASH WINDOW**. The effects of these parameters are explained throughout this Guide.
- Now the **EXPAND PULSES** may be activated, if necessary. In case of option -2, additional **BASELINE FILTER** adjustments are required.

- The **AMPLITUDE THRESHOLD**, **AREA/AMP. LOW THR.** and **AREA/AMP. HIGH THR.** should now be set. These are always applied to the highest point of the pulse and to the areas calculated by the pulse integration, regardless of the **AMPLITUDE OPTION** that is yet to be selected! These thresholds are 'relative' to the pulse polarity, meaning that they are applied in a sense:  $\text{polarity} \times \text{pulse} \geq \text{threshold}$ . (Otherwise, all pulses of one polarity would be either kept or rejected, while the thresholds would sensibly apply only to the other polarity.) Again, none of these thresholds should be so severe that true pulses are being discarded by them. But they may affect procedures such as the Pulse Shape fitting, so they do need to be appropriately set!
- Amplitude treatment may finally be selected via **AMPLITUDE OPTION** parameter. In case of the Pulse Shape fitting (option 2), the **NUMBER OF PULSE SHAPES** and **PULSE SHAPE ADDRESS** also need to be set. The number of addresses for the separate Pulse Shape files must be **exactly** equal to the **NUMBER OF PULSE SHAPES**.

# Little practical advices (from users' experience)

- Reducing the **STEP SIZE** – even at the price of worsening the signal-to-noise ratio in the derivative – can often help in resolving the pileups.
- **SIGNAL WIDTH LOW THR.** and **SIGNAL WIDTH HIGH THR.** are used for elimination only before the pulse expansion! Therefore, they should be adjusted with expansion deactivated (**EXPAND PULSES=0**).
- **SIGNAL WIDTH LOW THR.** should be adjusted looking at the pulses from pileup, since they will be **cut short** by a following pulse! If pulses from pileup are not resolved, the expansion procedure will likely merge them!
- Filtered baseline (**BASELINE OPTION=1**) is calculated after the pulse elimination based **only** on the pulse width limits. Therefore, the **BASELINE FILTER** should be adjusted with the other elimination conditions (**AMPLITUDE THRESHOLD**, **AREA/AMP. LOW THR.** and **AREA/AMP. HIGH THR.**) completely relaxed.
- Elimination conditions must not be so severe that the true pulses would be rejected! If some of the false pulses do pass into the final root-files, they can and **should** be eliminated during the later data analysis, based on the additional data available therein.

# Little practical advices (from users' experience)

- The **MIXED POLARITY** option has nothing to do with bipolar pulses! It refers to the changing polarity between the different pulses (some are positive, some are negative), **not** within the same pulse! Bipolar pulses are to be handled by the **EXPAND PULSES** option.
- When the **MIXED POLARITY**=1 is used in combination with Pulse Shape fitting (**AMPLITUDE OPTION**=2), it is crucial to adjust well the elimination parameters, since the intake of false pulse during pulse-by-pulse search may affect the recognition of the following pulses in line.
- In the root-files *polarity*=1 means 'regular' pulses, while *polarity*=-1 means 'inverted' pulses; **not** necessarily 'positive' or 'negative' ones.
- Without Pulse Shape fitting and with **MIXED POLARITY**=1 activated, initially reported pulses of opposing polarities will overlap. It is user's responsibility to eliminate the false candidates by the appropriate set of elimination conditions. If this is not enough, the further elimination may and **should** be performed during the later data analysis, based on the data available in the root-files. The amount of overlaps is reported in the version of the program with the graphical user interface.
- The **AMPLITUDE THRESHOLD** is 'relative' to the pulse polarity.

- Within **G-FLASH WINDOW** everything is protected from elimination so it is possible that some 'bad' pulses from there find their way into the final root-files. For example, some negative amplitudes may be found in spite of setting the positive amplitude threshold. Evidently, this does not mean that the signal analysis went wrong, but rather that the final elimination should be done during the later data analysis!
- If some pulses don't seem to be eliminated, while it is obvious that they should have been, then they are trapped within the **G-FLASH WINDOW**, where **everything** is protected against elimination. In this case the **G-FLASH WINDOW** should be checked and, if necessary, readjusted.
- Exceptionally, the pulses from  $\gamma$ -flash region **may** be eliminated in case of the Pulse Shape fitting, since the baseline will have already been calculated when the repeated search for pulses (one-by-one) starts. If the  $\gamma$ -flash is indeed so different from regular pulses that it prompts the elimination, then it is safer not to fit it, so that the subtraction of possibly exaggerated rebound would not affect the immediate  $\gamma$ -flash surroundings.

## Think creatively!

This is easily one of the most important advices one can receive regarding any kind of optimization. If one gets stuck around the **local** optimum and things seem to be as good as they can get – but not quite there yet – throwing the parameters completely off balance often works wonders for getting closer to the **global** optimum. In fact, the 'darkest corners' of the parameter space should always be intentionally checked, in order to get the feeling for procedures' inner workings and to confirm that the suspected optimal set of parameters is indeed optimal. This is also one of the best ways for identifying the errors, weak points or limitations in the procedures.

# Happy pulse hunting!

